| | | | | |
|---|---|---|---|---|
| *fraccap*, *fc* | | | | fractional capability variable |
| *expression*, $x$, $g$, $a$, $b$ | | | | expression variable |
| *integer*, $k$ | | | | integer variable |
| *float64*, *flt* | | | | 64-bit floating-point variable |
| *terminals* | ::= | | | |
| | \| | $\lambda$ | | |
| | \| | $\otimes$ | | |
| | \| | $\multimap$ | | |
| | \| | $\vdash$ | | |
| | \| | $\in$ | | |
| | \| | $\forall$ | | |
| | \| | Cap | | |
| | \| | Type | | |
| | \| | ! | | |
| $f$ | ::= | | | fractional capability |
| | \| | *fc* | | variable |
| | \| | **Zero** | | zero |
| | \| | **Succ** $f$ | | successor |
| $ti$ | ::= | | | non-linear type |
| | \| | **int** | | integer |
| | \| | $f_{64}$ | | 64-bit floats (doubles) |
| | \| | **bool** | | booleans |
| | \| | $t \multimap t'$ | | arrow (multiple-use) |
| $t$ | ::= | | | linear type |
| | \| | 1 | | unit |
| | \| | !($ti$) | | multiple-use type |
| | \| | $t \otimes t'$ | | pair |
| | \| | $t \multimap t'$ | | arrow (single-use) |
| | \| | $\forall fc.t$ | bind *fc* in $t$ | frac. cap. abstraction |
| | \| | **Arr** $[f]$ | | array |
| | \| | $t\{f/fc\}$ | M | substitution |
| $p$ | ::= | | | primitive |
| | \| | **set** | | array index assignment |
| | \| | **get** | | array indexing |
| | \| | **add** | | 64-bit float addition |
| | \| | **sub** | | 64-bit float subtraction |
| | \| | **mul** | | 64-bit float multiplication |
| | \| | **div** | | 64-bit float division |
| | \| | **eq** | | 64-bit float equality |
| | \| | **lt** | | 64-bit float less-than |
| | \| | **iadd** | | integer addition |
| | \| | **isub** | | integer subtraction |
| | \| | **imul** | | integer multiplication |
| | \| | **idiv** | | integer division |
| | \| | **ieq** | | integer equality |
| | \| | **ilt** | | integer comparsion (less-than) |
| | \| | **and** | | boolean conjuction |

| | | | |
|---|---|---|---|
| | \| | **or** | boolean disjunction |
| | \| | **not** | boolean negation |
| | \| | **split_perm** | share array |
| | \| | **merge_perm** | unshare array |
| | \| | **free** | free arrary |
| | \| | **copy** | copy array |
| | \| | **swap** | swap array |
| | \| | **asum** | $\sum_i \lvert x_i \rvert$ |
| | \| | **axpy** | $x := \alpha x + y$ |
| | \| | **dot** | $x \cdot y$ |
| | \| | **nrm2** | $\lVert x \rVert^2$ |
| | \| | **rot** | plane rotation |
| | \| | **rotg** | Givens rotation |
| | \| | **rotm** | modified givens rotation |
| | \| | **rotmg** | generate modified Givens rotation |
| | \| | **scal** | $x := \alpha x$ |
| | \| | **amax** | index of maximum absolute value |

| | | | | |
|---|---|---|---|---|
| $e$ | ::= | | | expression |
| | \| | $x$ | | variable |
| | \| | $k$ | | integer |
| | \| | *flt* | | 64-bit floating-point |
| | \| | $()$ | | unit introduction |
| | \| | $\mathbf{let}\,() = e; e'$ | | unit elimination |
| | \| | **true** | | true (boolean introduction) |
| | \| | **false** | | false (boolean introduction) |
| | \| | $\mathbf{if}\ e\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2$ | | if (boolean elimination) |
| | \| | $(e, e')$ | | pair introduction |
| | \| | $\mathbf{let}\,(a, b) = e; e'$ | bind $a \cup b$ in $e'$ | pair elimination |
| | \| | $\lambda x : t.e$ | bind $x$ in $e$ | abstraction |
| | \| | $\mathbf{fix}\, g : !(t \multimap t') = e$ | bind $g$ in $e$ | fixpoint operator |
| | \| | $e\ e'$ | | application |
| | \| | $\mathbf{Array}\ e$ | | array introduction |
| | \| | $\mathbf{let}\, x = e; e'$ | bind $x$ in $e'$ | array elimination |
| | \| | $p$ | | Level 1 BLAS routine primitives |
| | \| | $\forall\, fc.e$ | | frac. cap. abstraction |
| | \| | $e[f]$ | | frac. cap. specialisation |

| | | | |
|---|---|---|---|
| $\Theta$ | ::= | | fractional capability environment |
| | \| | . | |
| | \| | $\Theta, fc$ | |

| | | | |
|---|---|---|---|
| $\Gamma$ | ::= | | linear types environment |
| | \| | . | |
| | \| | $\Gamma, x : t$ | |
| | \| | $\Gamma, \Gamma'$ | |

| | | | |
|---|---|---|---|
| $\Delta$ | ::= | | linear types environment |
| | \| | . | |
| | \| | $\Delta, x : ti$ | |
| *formula* | ::= | | |
| | \| | *judgement* | |
| | \| | $x : ti \in \Delta$ | |
| | \| | $x : t \in \Gamma$ | |
| | \| | $fc \in \Theta$ | |
| *Well_Formed* | ::= | | |
| | \| | $\Theta \vdash f\ \mathsf{Cap}$ | Valid fractional capabilities |
| | \| | $\Theta \vdash t\ \mathsf{Type}$ | Valid types |
| *Types* | ::= | | |
| | \| | $\Theta; \Delta; \Gamma \vdash e : t$ | Tying rules for expressions (no primitives yet) |
| *judgement* | ::= | | |
| | \| | *Well_Formed* | |
| | \| | *Types* | |
| *user_syntax* | ::= | | |
| | \| | *fraccap* | |
| | \| | *expression* | |
| | \| | *integer* | |
| | \| | *float64* | |
| | \| | *terminals* | |
| | \| | *f* | |
| | \| | *ti* | |
| | \| | *t* | |
| | \| | *p* | |
| | \| | *e* | |
| | \| | $\Theta$ | |
| | \| | $\Gamma$ | |
| | \| | $\Delta$ | |
| | \| | *formula* | |

$\boxed{\Theta \vdash f\ \mathsf{Cap}}$ Valid fractional capabilities

$$\frac{fc \in \Theta}{\Theta \vdash fc\ \mathsf{Cap}}\ \text{WF\_Cap\_Var}$$

$$\frac{}{\Theta \vdash \mathbf{Zero}\ \mathsf{Cap}}\ \text{WF\_Cap\_Zero}$$

$$\frac{\Theta \vdash f\ \mathsf{Cap}}{\Theta \vdash \mathbf{Succ}\ f\ \mathsf{Cap}}\ \text{WF\_Cap\_Succ}$$

$\boxed{\Theta \vdash t\ \mathsf{Type}}$ Valid types

$$\frac{}{\Theta \vdash 1\ \mathsf{Type}}\ \text{WF\_Type\_Unit}$$

$$\frac{}{\Theta \vdash !(\mathbf{int})\ \mathsf{Type}}\ \text{WF\_Type\_Int}$$

$$\frac{}{\Theta \vdash !(f_{64})\,\mathsf{Type}}\ \textsc{WF_Type_Float64}$$

$$\frac{}{\Theta \vdash !(\mathbf{bool})\,\mathsf{Type}}\ \textsc{WF_Type_Bool}$$

$$\frac{\Theta \vdash t\,\mathsf{Type} \quad \Theta \vdash t'\,\mathsf{Type}}{\Theta \vdash t \otimes t'\,\mathsf{Type}}\ \textsc{WF_Type_Pair}$$

$$\frac{\Theta \vdash t\,\mathsf{Type} \quad \Theta \vdash t'\,\mathsf{Type}}{\Theta \vdash t \multimap t'\,\mathsf{Type}}\ \textsc{WF_Type_Lollipop}$$

$$\frac{\Theta \vdash t \multimap t'\,\mathsf{Type}}{\Theta \vdash !(t \multimap t')\,\mathsf{Type}}\ \textsc{WF_Type_Fixt}$$

$$\frac{\Theta \vdash f\,\mathsf{Cap}}{\Theta \vdash \mathbf{Arr}\,[f]\,\mathsf{Type}}\ \textsc{WF_Type_Array}$$

$$\frac{\Theta, fc \vdash t\,\mathsf{Type}}{\Theta \vdash \forall\, fc.t\,\mathsf{Type}}\ \textsc{WF_Type_ForAll}$$

$\boxed{\Theta; \Delta; \Gamma \vdash e : t}$ Tying rules for expressions (no primitives yet)

$$\frac{}{\Theta; \Delta; \cdot, x : t \vdash x : t}\ \textsc{Ty_Var}$$

$$\frac{x : ti \in \Delta}{\Theta; \Delta; \Gamma \vdash x : !(ti)}\ \textsc{Ty_Var_Bang}$$

$$\frac{}{\Theta; \Delta; \Gamma \vdash k : !(\mathbf{int})}\ \textsc{Ty_Int_Intro}$$

$$\frac{}{\Theta; \Delta; \Gamma \vdash \mathit{flt} : !(f_{64})}\ \textsc{Ty_Float64_Intro}$$

$$\frac{}{\Theta; \Delta; \cdot \vdash () : 1}\ \textsc{Ty_Unit_Intro}$$

$$\frac{\Theta; \Delta; \Gamma \vdash e : 1 \quad \Theta; \Delta; \Gamma' \vdash e' : t}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{let}\,() = e; e' : t}\ \textsc{Ty_Unit_Elim}$$

$$\frac{}{\Theta; \Delta; \Gamma \vdash \mathbf{true} : !(\mathbf{bool})}\ \textsc{Ty_Bool_True}$$

$$\frac{}{\Theta; \Delta; \Gamma \vdash \mathbf{false} : !(\mathbf{bool})}\ \textsc{Ty_Bool_False}$$

$$\frac{\Theta; \Delta; \Gamma \vdash e : !(\mathbf{bool}) \quad \Theta; \Delta; \Gamma_1 \vdash e_1 : t \quad \Theta; \Delta; \Gamma_2 \vdash e_2 : t}{\Theta; \Delta; \Gamma, \Gamma_1, \Gamma_2 \vdash \mathbf{if}\, e\, \mathbf{then}\, e_1\, \mathbf{else}\, e_2 : t}\ \textsc{Ty_Bool_Elim}$$

$$\frac{\Theta; \Delta; \Gamma \vdash e : t \quad \Theta; \Delta; \Gamma' \vdash e' : t'}{\Theta; \Delta; \Gamma, \Gamma' \vdash (e, e') : t \otimes t'}\ \textsc{Ty_Pair_Intro}$$

$$\frac{\Theta; \Delta; \Gamma \vdash e_{12} : t_1 \otimes t_2 \quad \Theta; \Delta; \Gamma', a : t_1, b : t_2 \vdash e : t}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{let}\,(a, b) = e_{12}; e : t}\ \textsc{Ty_Pair_Elim}$$

$$\frac{\Theta \vdash !(ti)\ \mathsf{Type} \quad \Theta; \Delta, x : ti; \Gamma \vdash e : t}{\Theta; \Delta; \Gamma \vdash \lambda x : !(ti).e : !(ti) \multimap t}\ \textsc{Ty\_Lambda\_Bang}$$

$$\frac{\Theta \vdash t'\ \mathsf{Type} \quad \Theta; \Delta; \Gamma, x : t' \vdash e : t}{\Theta; \Delta; \Gamma \vdash \lambda x : t'.e : t' \multimap t}\ \textsc{Ty\_Lambda}$$

$$\frac{\Theta; \Delta; \Gamma \vdash e : t' \multimap t \quad \Theta; \Delta; \Gamma' \vdash e' : t'}{\Theta; \Delta; \Gamma, \Gamma' \vdash e\ e' : t}\ \textsc{Ty\_App}$$

$$\frac{\Theta; \Delta; \Gamma \vdash e : !(\mathbf{int})}{\Theta; \Delta; \Gamma \vdash \mathbf{Array}\ e : \mathbf{Arr}\ [\mathbf{Zero}]}\ \textsc{Ty\_Array\_Intro}$$

$$\frac{\Theta; \Delta; \Gamma \vdash e : \mathbf{Arr}\ [f] \quad \Theta; \Delta; \Gamma', x : \mathbf{Arr}\ [f] \vdash e' : t'}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{let}\ x = e; e' : t'}\ \textsc{Ty\_Array\_Elim}$$

$$\frac{\Theta, fc; \Delta; \Gamma \vdash e : t}{\Theta; \Delta; \Gamma \vdash \forall\, fc.e : \forall\, fc.t}\ \textsc{Ty\_ForAll\_frac\_cap}$$

$$\frac{\Theta \vdash f\ \mathsf{Cap} \quad \Theta; \Delta; \Gamma \vdash e : \forall\, fc.t}{\Theta; \Delta; \Gamma \vdash e[f] : t\{f/fc\}}\ \textsc{Ty\_Spec\_frac\_cap}$$

$$\frac{\Theta \vdash t \multimap t'\ \mathsf{Type} \quad \Theta; \Delta, g : t_1 \multimap t_2; \cdot \vdash e : t_1 \multimap t_2}{\Theta; \Delta; \Gamma \vdash \mathbf{fix}\ g : !(t \multimap t') = e : !(t \multimap t')}\ \textsc{Ty\_Fix}$$

```
Definition rules:          19 good    0 bad
Definition rule clauses: 43 good    0 bad
```